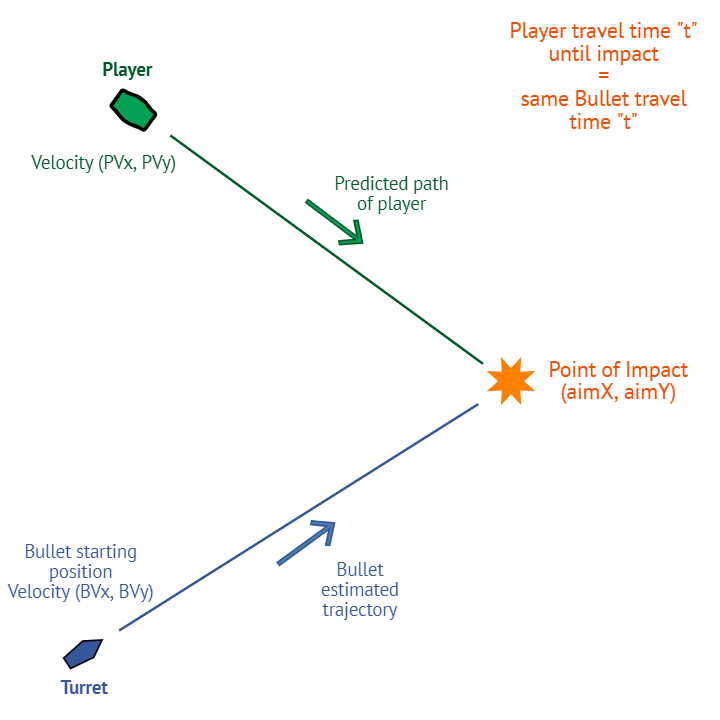
**Hitting a Moving Target**



Aim

A 2D top down game with enemy AI needs a targeting system to shoot at the player. The shots are not instant so the system will need to aim ahead of the player's current position. Shots aimed at the player should hit if the player is not moving, or if the player is moving at a constant speed. To avoid hits, the player must need to change while the shot is travelling. As input to the system you will be given the player's relative location and velocity. You must return the relative clockwise angle in degrees needed to hit the player.

Approach

When the turret fire, it will calculate the point of impact with the player. The distance between the player and the bullet will reach 0 in the same time “t”. The only unknown parameter is this time “t” that we have to find. At the moment of fire, the system knows the following information:

* *Player:* Initial position (player.position.x, player.position.y). Moves in linear velocity noted (Player.velocity.x, Player.velocity.y)
* *Bullet:* Initial position (transform.position), bullet speed (shotSpeed), linear velocity.

Getting to the equation

We need to create two equations, one for the player and one for the bullet, then mixing them to get the same unknown parameter. We note:

Bullet position (Bx, By) and Bullet velocity (BVx, BVy), Bullet Speed (s)

Player position (Px, Py) and Player Velocity (PVx, PVy)

Time to collision (t)

Collision Point (Cx, Cy)

The distance D between the bullet and the player can be calculated:

* D2 = (Cx – Bx)2 + (Cy – By)2

Also Distance = Speed \* Time, written as D = s \* t

Equating both, we get:

s2 \* t2 = (Cx – Bx)2 + (Cy – By)2

The player will reach the same point (Cx, Cy) during the same time (t). We can find this point by:

Cx = t \* PVx +Px

Cy = t \* PVy + Py

Substituting those in the previous equations gives us:

s2 \* t2 = ((t \* PVx +Px) – Bx)2 + ((t \* PVy + Py) – By)2

Expanding the equation

We need the equation to be quadratic, under the form ax2 + bx + c = 0

By expanding the previous equation, we get:

(PVx2 + PVy2 – s2) \* t2 + 2 \* (PVx \* (Px – Bx) + PVy \* (Py – By)) \* t + (Py – By)2 + (Px – Bx)2 = 0

Finally, we have our final quadratic equation.

We can calculate:

a = (PVx2 + PVy2 – s2)

b = 2 \* (PVx \* (Px – Bx) + PVy \* (Py – By))

c = (Py – By)2 + (Px – Bx)2

In Unity Terms, in my case

Px = player.gameObject.transform.position.x

Py = player.gameObject.transform.position.y

PVx = player.velocity.x

PVy = player.velocity.y

Bx = transform.position.x

By = transform.position.y

s = shotSpeed

The resolution of the equation and how we use it is explained in comments in the script.

How to test

In the scene, you can control the player using WASD. The turret will calculate automatically the angle needed to hit the player. Press Spacebar to shoot a bullet from the turret.